3-17-2005

Phylogenetic Compression and Model Selection: An Improved Encoding Scheme

Cécile Ané
Iowa State University

Oliver Eulenstein
Iowa State University, oeulenst@iastate.edu

Raul Piaggio-Talice
Iowa State University

Phylogenetic Compression and Model Selection: An Improved Encoding Scheme

Abstract
A central problem in phylogenetics is to select a hypothesis that best describes the evolutionary history leading to the sequences in a given alignment. Ané and Sanderson recently proposed a method of making this decision by selecting the hypothesis that minimizes the code length for the hypothesis plus the code length for the alignment given the hypothesis. In this work, we present an improvement in the coding scheme that results in shorter codes and briefly analyze the differences in performance with the original encoding.

Disciplines
Theory and Algorithms

Phylogenetic Compression and Model Selection: An Improved Encoding Scheme (TR05-07)

Cécile Ané*, Oliver Eulenstein†, Raul Piaggio-Talice†‡

March 17, 2005

Abstract

A central problem in phylogenetics is to select a hypothesis that best describes the evolutionary history leading to the sequences in a given alignment. Ané and Sanderson recently proposed [1] a method of making this decision by selecting the hypothesis that minimizes the code length for the hypothesis plus the code length for the alignment given the hypothesis. In this work, we present an improvement in the coding scheme that results in shorter codes and briefly analyze the differences in performance with the original encoding.

1 Introduction

A central problem in phylogenetics is to select a hypothesis that best describes the evolutionary history leading to the sequences in a given alignment. Ané and Sanderson recently proposed [1] a method of making this decision by using the minimum description length principle from algorithmic information theory [2]. In this approach, the alignment is described by two-part encoding composed of the code for a hypothesis plus the code for the alignment using such hypothesis. The hypothesis assumed to be correct will be the one that minimizes such encoding.

Finding the encoding that realizes the minimum description length of an alignment also provides an efficient way of compressing sets of homologous sequences. Over the past few years, the number of stored DNA sequences has grown dramatically. GenBank alone has experimented an exponential growth, having reached by the end of 2004 more than 40 million sequences comprising more than 44.5 billion bases (ftp://ftp.ncbi.nih.gov/genbank/gbrel.txt). This

*Dept. of Statistics, University of Wisconsin-Madison, ane@stat.wisc.edu
†Dept. of Computer Science, Iowa State University, [oeulenst,rpiaggio]@cs.iastate.edu
‡Corresponding author.

growth rate is not expected to slow down in the near future. Therefore, DNA compression techniques may be useful in this context too.

In this work, we extend the results from [1] by providing an improved encoding mechanism. Since the new mechanism results in shorter encodings closer to the minimum description length, it is expected to allow us to sharpen the decision criterion as well as providing a better compression mechanism. Our proposed method is still computable despite the fact that finding the hypothesis that minimizes a two-part code is akin to computing the Kolmogorov complexity of the alignment, known to be uncomputable [2].

In the next section we describe the new encoding mechanism. In Section 3 we reevaluate the analyses from [1] with the new method. These empirical results show the extent of improvement by compressing alignments with the new method while suggesting it still retains the ability to discern between conflicting hypotheses. Finally, in Section 4, we discuss implications and future work. The proof of the improvement in encoding is presented in the Appendix.

2 Encoding

2.1 Definitions

We address the problem of compressing a sequence alignment consisting of n sequences of length k over the alphabet {A, C, G, T} and with no gaps. This is a simplification of the cases encountered in real life, but the general approach can be extended to contemplate irregularities, as discussed later.

Given an odd number $m \in \mathbb{N}$, we will use the standard definition of *double factorial*

$$m!! = m(m-2)(m-4)...(3)(1) = \frac{m!}{2^{\frac{m-1}{2}}\left(\frac{m-1}{2}\right)!}.$$

It is a well-known fact that there are $(2n-5)!!$ possible unrooted binary phylogenetic trees over n leaves.

Given a number $m \in \mathbb{N}$, the *description length* $d(m)$ of m is a function $d : \mathbb{N} \to \mathbb{N}$ that transforms a number to the number of bits needed to encode it in a self-delimiting way. One possible way to achieve this is by using a logarithmic ramp as described in [2], which results for example in $d(m) = \lceil \log_2 m \rceil + \lceil \log_2(\lceil \log_2 m \rceil) \rceil + ... + 2 + 1$ when m is not a power of 2.

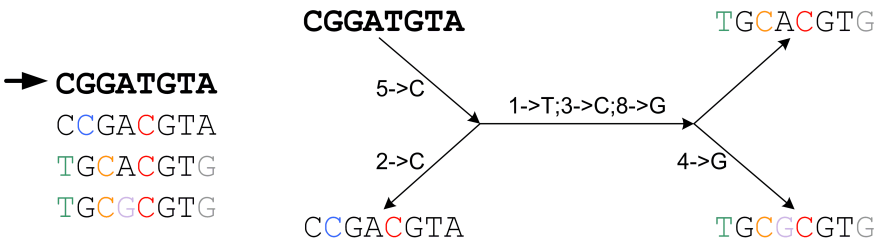

Figure 1: Example of encoding and reconstructing the alignment on the left. The first sequence is chosen as the initial sequence and the changes are encoded across a most parsimonious tree.

2.2 Compression method

Ané and Sanderson's method (Figure 1) consisted of initially encoding an unrooted binary tree over n leaves labeled each with one sequence from the alignment. If we explicitly describe one of the sequences (called the *initial sequence*) and then describe the changes that occur across the tree, we can reconstruct the sequence on each leaf by traversing the tree.This recovers the whole set of sequences that form the alignment. The encoding used by Ané and Sanderson resulted in an encoding length of

$$\underbrace{\underbrace{\underbrace{2n-4+n\lceil\log_2 n\rceil}_{Tree}+\underbrace{2k}_{Init.\ seq.}}_{Part\ 1:\ Alignment\ model}+\underbrace{2k+(2+\lceil\log_2 2n-3\rceil)L+\lceil\log_2 2n-3\rceil}_{Part\ 2:\ Alignment\ description}}$$

bits, where L is the total number of changes across the tree (parsimony score). Part 1 is the length of encoding the tree plus the length of encoding the initial sequence, and Part 2 is the length of encoding the changes. The actual values of n, k and L can be recovered from the encoding and need not be explicitly described. It is easy to see that choosing the most parsimonious tree yields the most compression.

The new scheme follows the same general principle but the encoding is carried out in a way that results in a shorter code. The general approach is to use index-based two-part codes. This means that whenever we describe a model that defines a finite set of elements, we then identify a particular element by agreeing on an enumeration of the elements of the model and then providing the index of the particular element in such an enumeration. This can be done both to identify an unrooted binary phylogenetic tree with a given number of leaves and to identify a particular pattern of changes across a tree (Figure 2).

The initial sequence can still be encoded by using $2k$ bits. The encoding of the tree is done by providing the number n followed by the index of the

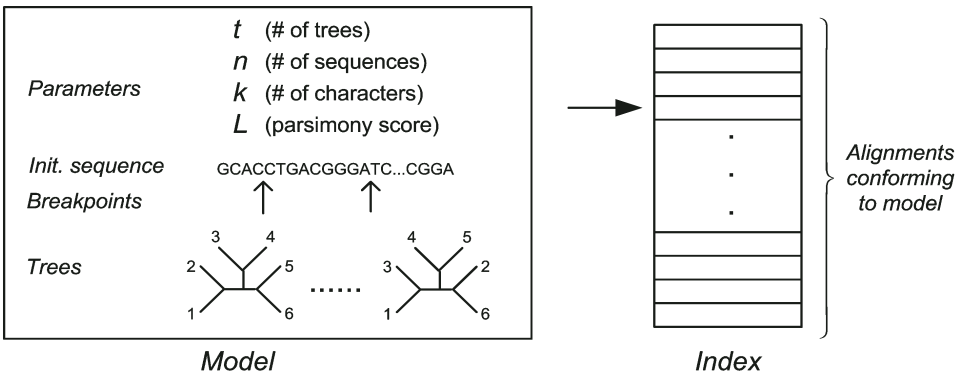

Figure 2: Example of two-part code, by encoding the model (left) that defines a finite set of possible alignments and the index (right) of the encoded alignment. In the encoding we consider for one tree ($t = 1$), we don't include t (or any breakpoints).

particular tree in an enumeration of the $(2n-5)!!$ existing unrooted binary phylogenetic trees. This takes $d(n) + \lceil \log(2n-5)!! \rceil$ bits. Together with k and L, which can be encoded in $d(k) + d(L)$ bits, the model for the alignment is completely described.

The final part of the encoding identifies the particular alignment among all the ones that conform to the model parameters. We can perform this encoding by enumerating all the ways to combine L changes in the tree and providing the index of the set of changes that will result in the original alignment when traversing the tree from the initial sequence. For each change, we must identify:

- The branch of the tree where it occurs.
- The character (position in the sequence) that changes.
- The nature of the change. That is, the new nucleotide after the change.

We know the tree has $2n-3$ edges and that there cannot be more than one change per character in each edge, therefore there are $k(2n-3)$ possible ways of combining characters and edges of the tree. This results in $\binom{k(2n-3)}{L}$ possible ways of choosing L changes in k positions and in the $2n-3$ branches of the tree.

In order to encode the nature of each change note that, since there are four nucleotides, there are, given the previous nucleotide and the fact that there is a change, only three possible types of change. For example, we could convene the three types of changes to be: transition (A↔G,C↔T), tranversion into complementary nucleotide (A↔T,C↔G) or tranversion into non-complementary nucleotide (A↔C,G↔T). Therefore, the space of all possible combinations of L types of change will have size 3^L.

We can then conclude that $\left\lceil \log_2 3^L \binom{k(2n-3)}{L} \right\rceil$ bits are enough to identify a particular set of changes across the tree and thus recover the alignment. Therefore, the length of the encoded alignment using the new scheme is

$$\underbrace{\underbrace{\underbrace{d(k) + d(L) + d(n)}_{Parameters} + \underbrace{\lceil \log_2 (2n-5)!! \rceil}_{Tree\ index} + \underbrace{2k}_{\substack{Init.\\ seq.}}}_{Part\ 1:\ Alignment\ model} + \underbrace{\left\lceil \log_2 3^L \binom{k(2n-3)}{L} \right\rceil}_{Part\ 2:\ Alignment\ index}}$$

bits. The Appendices contain the proof that this encoding is shorter than the original one whenever $k \geq 7$ and $n \geq 3$. Also note that, as in the original case, this formula is minimized when L is minimal. That is, when the most parsimonious tree is used. Note that in both cases the initial sequence is uncompressed. We could compress it with a regular method, like GenCompress [3] and then these equations would express upper bounds.

2.2.1 Multiple trees

The encoding scheme can be extended to cases in which different trees are used for different partitions of the alignment. This is especially important since it is what will allow the confrontation of hypotheses consisting of one evolutionary history against those consisting of multiple ones.

As done in the original work [1], we consider the case in which we have t contiguous partitions of the alignment. In this case, we need to encode these additional information:

- the value of t,
- the additional $t-1$ trees,
- the $t-1$ breakpoints between the partitions.

This yields an encoded length of

$$\underbrace{d(n) + d(t) + d(k) + d(L)}_{Parameters} + \underbrace{t\lceil \log_2 (2n-5)!! \rceil}_{Tree\ indices}$$
$$+\underbrace{(t-1)\lceil \log_2 k \rceil}_{Breakpoints} + \underbrace{2k}_{\substack{Init.\\ seq.}} + \underbrace{\left\lceil \log_2 3^L \binom{k(2n-3)}{L} \right\rceil}_{Alignment\ index}$$

bits, where L now denotes the sum of parsimony scores of all the trees. Note that by providing the breakpoints we can know which tree to use for a particular

change given its position in the sequence. Therefore, the size of the set defined by the model (and therefore the number of bits needed for the alignment index) does not change with respect to the parameters. However, the value of L will most probably change, and the size of the index will change with it.

As shown in the Appendix, the use of an index results in a shorter encoding for a tree than in the original method. However, the additional method only needs $t \log_2(2n-3)$ additional bits to describe the breakpoints, which may prove shorter than the $(t-1)\lceil \log_2 k \rceil$ bits needed in this case. But this difference is usually a few bits and easily compensated by the gain in the rest of the encoding. Consider the case of $n = 16$ sequences of length $k = 10^6$. In this case, $\log_2(2n-3) - \lceil \log_2 k \rceil = 16$. This is compensated by the encoding of the first tree, in which we save at least $2n - 6 = 26$ bits (see Appendix).

An alternative way to encode the set of trees, also proposed by Ané & Sanderson [1], is to use Nearest-Neighbor Interchange (NNI) operations to encode the set of trees. In this approach, only the first tree is explicitly encoded. Each subsequent tree is described by detailing the NNI operations needed to obtain it from the previous one. Since there are only two possible NNI operations over each internal edge of an unrooted binary phylogenetic tree, each NNI operation can be encoded in $\lceil \log_2(n-3) \rceil + 1$ bits. Let δ_i be the NNI distance from tree $i-1$ to tree i and $\overline{\delta} = \left(\sum_{i=2}^{t} \delta_i \right) / (t-1)$ its average value, then the resulting code length is

$$\underbrace{d(n) + d(t) + d(k) + d(L)}_{Parameters} + \underbrace{\lceil \log_2 (2n-5)!! \rceil}_{Tree\ indices}$$
$$+ \underbrace{\sum_{i=2}^{t} d(\delta_i) + (t-1)\overline{d}\,(\lceil \log_2(n-3) \rceil + 1)}_{NNI\ operations}$$
$$+ \underbrace{(t-1)\lceil \log_2 k \rceil}_{Breakpoints} + \underbrace{2k}_{\substack{Init. \\ seq.}} + \underbrace{\left\lceil \log_2 3^L \binom{k(2n-3)}{L} \right\rceil}_{Alignment\ index}$$

bits. If the trees are fairly similar, this may result in a shorter encoding than when describing each tree independently.

3 Real-world data

We applied our compression method on the dataset that was used in [1]. The dataset consists of 638 clusters of homologous green-plant protein coding genes, and the details of extraction from GenBank can be found in [4]. The minimum gain in the length of the encoding was of 8.79% while the maximum gain was of

	Original (Gain)	New (Gain)
Total evidence	50083	42010
1,2 + 3 forest (separate trees)	49987 (+96)	41879 (+131)
1,2 + 3 forest (NNI)	49880 (+203)	41856 (+154)
psaA + psaB forest (separate trees)	50197 (−114)	42068 (−58)
psaA + psaB forest (NNI)	50086 (−3)	42045 (−35)

Table 1: Compressed lengths of alignment containing psaA and psaB genes for 19 land plants using the original and the new method.

49.02%, with an average of 26.18% and a standard deviation of 10.50%. In most cases the original method already performed better than GenCompress [3].

We also applied the new method in the case study of conflicting signals from [1]. The results are shown in Table 1. In this case, the alignment contains two plastid genes, *psaA* and *psaB*, for 19 land plants, originally studied in [5]. Using the original method it was concluded that using different trees for each gene did not provide a better encoding than using a total evidence tree. However, using one tree for the first and second codon positions and another for the third did improve the encoding length due to varying rates of evolution. Results comparing both methods are summarized in Table 1, showing the gain in each case that uses two trees as the improvement from the total evidence approach. Results are presented for explicit encoding of the trees and for NNI distance encoding of additional trees. We can see that in all cases the new method provides a better compression, and that the margins are similar.

The data suggests that the best hypothesis among the ones considered is one consisting of partitions according to codon positions. The method described does not contemplate this case since it assumed the partitions to be contiguous. However, this partitioning scheme has low algorithmic complexity and so the method could be easily adapted to it without negatively affecting the code length too much. In fact, it could even mean a further decrease in the number of bits needed: the description of codon positions is bounded by a constant, whereas breakpoint descriptions grow with k.

4 Conclusion/Future work

The preliminary experiments presented here allow us to conclude that for compression purposes the new method presents a significant improvement. The new method also seems to preserve the ability to discern between conflicting hypotheses. However, there doesn't seem to be a significant improvement when used for this purpose.

All these results are based on bounds derived from equations. If we are to actually use the new encoding to perform compression, there are still challenges to be solved. By contrast to the original method, an implementation of the new method is not straightforward. However, such an implementation can be achieved by ranking and unranking of unrooted binary phylogenetic trees and of fixed-size subsets of a given set. The study of effective methods to accomplish these tasks is the subject of current work. Furthermore, as mentioned before, any practical implementation should compress the initial sequence by using some other technique.

Furthermore, the code we use contains redundancies. It is a well-known fact that for a particular character there may be several ways of distributing the minimal number of changes on a tree such that they result in the same set of states on the leaves. Detecting redundancies of this kind and eliminating them from the code will therefore improve the compression. As an additional constraint, it is imperative not to degrade the running time of the method by doing this.

Acknowledgements

We thank Mike Sanderson for his support and for providing a creative environment with valuable discussions of the ideas presented in this work.

References

[1] C. Ané and M. J. Sanderson, "Missing the forest for the trees: Phylogenetic compression and its implications for inferring complex evolutionary histories," *Systematic Biology*, 2005 (To Appear).

[2] M. Li and P. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications*, revised and expanded second edition 1997 ed. New York: Springer-Verlag, 1993.

[3] X. Chen, M. Li, B. Ma, and J. Tromp, "Dnacompress: fast and effective dna sequence compression," *Bioinformatics (oxford)*, vol. 18, pp. 1696–1698, 2002.

[4] M. J. Sanderson, A. C. Driskell, O. Eulenstein, and S. Langley, "Obtaining maximal concatenated phylogenetic data sets from large sequence databases," *Molecular Biology and Evolution*, vol. 20, pp. 1036–1042, 2003.

[5] M. J. Sanderson, J. M. Wojciechowski, M. F.and Hu, T. S. Khan, and S. G. Brady, "Error, bias, and long-branch attraction in data for two chloroplast photosystem genes in seed plants," *Molecular Biology and Evolution*, vol. 17, pp. 782–797, 2000.

Appendix: Encoding Improvements

Encoding of a Tree

In order to prove the improvement in the encoding, we define a function $f(n)$ as the difference of the bits needed to encode a tree between the old method and the new one. We prove that the difference is positive and that it increases as n increases. Throughout the proofs we use ln for $\log_e$ and log for $\log_2$.

Proposition 1 *Let* $f(n) = 2n - 4 + n\lceil \log n \rceil - \lceil \log((2n-5)!!) \rceil$. *If* $n \geq 3$ *then* $f(n) > 2n - 6 \geq 0$.

Proof. We know that $n \ln n - n < \ln(n!) < (n+1)\ln(n+1) - n$ (Stirling inequality). Then, $n \log n - \frac{n}{\ln 2} < \log(n!) < (n+1)\log(n+1) - \frac{n}{\ln 2}$ and so

$$\begin{aligned}
\log((2n-5)!!) &= \log \frac{(2n-5)!}{2^{n-3}(n-3)!} \\
&= \log((2n-5)!) - (n-3) - \log((n-3)!) \\
&< (2n-4)\log(2n-4) - \frac{(2n-5)}{\ln 2} \\
&\quad -(n-3) - (n-3)\log(n-3) + \frac{(n-3)}{\ln 2} \\
&= (2n-4)\log(n-2) + (2n-4) \\
&\quad -(n-3) - (n-3)\log(n-3) - \frac{n-2}{\ln 2} \\
&= (2n-4)\log(n-2) - (n-3)\log(n-3) - \frac{n-2}{\ln 2} + n - 1.
\end{aligned}$$

Therefore,

$$\begin{aligned} f(n) &= 2n-4+n\lceil\log n\rceil-\lceil\log((2n-5)!!)\rceil \\ &> 2n-4+n\lceil\log n\rceil-\log((2n-5)!!)-1 \\ &> n\lceil\log n\rceil-(2n-4)\log(n-2) \\ &\quad +(n-3)\log(n-3)+\frac{n-2}{\ln 2}+n-4 \\ &= \underbrace{n\lceil\log n\rceil-n\log(n-2)}_{g(n)} \\ &\quad +\underbrace{(n-3)\log(n-3)-(n-4)\log(n-2)}_{h(n)}+\left(\frac{1+\ln 2}{\ln 2}\right)n-2\left(\frac{1+2\ln 2}{\ln 2}\right). \end{aligned}$$

We can easily see that $g(n)\geq 0$. For $h(n)$ consider

$$h'(n)=\frac{1}{\ln 2}\left(\frac{2}{n-2}+\ln\left(\frac{n-3}{n-2}\right)\right).$$

Then,

$$\begin{aligned} h''(n) &= \frac{1}{\ln 2}\left(\frac{1}{n-3}-\frac{1}{n-2}-\frac{2}{(n-2)^2}\right) \\ &= \frac{1}{\ln 2}\left(\frac{1}{(n-2)(n-3)}-\frac{2}{(n-2)^2}\right) \\ &= \frac{1}{\ln 2}\left(\frac{(n-2)-2(n-3)}{(n-2)^2(n-3)}\right) \\ &= \frac{1}{\ln 2}\left(\frac{4-n}{(n-2)^2(n-3)}\right). \end{aligned}$$

We can see then that $h''(n)\leq 0$ when $n\geq 4$. Therefore, since $\lim_{n\to\infty} h'(n)=0$, we have that $h'(n)\geq 0$ when $n\geq 4$. Since $h(4)=0$, then when $n\geq 4$ we have that $h(n)\geq 0$ and so $f(n)>\left(\frac{1+\ln 2}{\ln 2}\right)n-2\left(\frac{1+2\ln 2}{\ln 2}\right)>2n-6$, as required.

When $n=3$ we have $f(n)=5>2n-6=0$. Therefore $f(n)>2n-6$ for all $n\geq 3$, which completes the proof. ■

4.1 Encoding of the Changes

We use a similar technique for the case of the encoding of the changes, by proving that the difference between both encodings is non-negative. We first prove two partial results.

Proposition 2 *For all $k \geq 1$, $n \geq 2$ and $0 \leq L \leq (2n-3)k$ we have*

$$\begin{aligned} & 4k + (2 + \log(2n-3))L \\ & - \left\{ 2k + \log(3)L - L\log\left(\frac{L}{(2n-3)k}\right) - ((2n-3)k - L)\log\left(\frac{(2n-3)k - L}{(2n-3)k}\right) \right\} \\ \geq \quad & 0.91k. \end{aligned}$$

The same inequality also holds when the first part is replaced by

$$4k + (2 + \lceil \log(2n-3) \rceil)L\,.$$

Proof. Let us call D the left hand part of the previous inequality. We introduce a change of variables by defining $p = L/k$. Note that $0 \leq p \leq 2n-3$ by hypothesis. We define a function f in terms of p and n such we have have $D = kf(p,n)$:

$$f(p,n) = 2 + (2 - \log(3))p + p\log p + (2n - 3 - p)\log\left(\frac{2n-3-p}{2n-3}\right).$$

We now need to show that the function f is bounded below by 0.91. We start by fixing n and finding the minimum of f when p varies between 0 and $2n-3$. The derivative

$$\frac{\partial f}{\partial p} = 2 - \log(3) + \log(p) + 1/\ln(2) - \log\left(\frac{2n-3-p}{2n-3}\right) - 1/\ln(2)$$

is equal to 0 if and only if

$$\log(p) - \log\left(\frac{2n-3-p}{2n-3}\right) = \log(3) - 2$$

i.e. when $p = p_0(n) = \dfrac{6n-9}{8n-9}$. For a fixed n, the function $f(p,n)$ is decreasing when $p \in [0, p_0]$ and increasing when $p \in [p_0, 2n-3]$. Therefore, its minimum is

$$g(n) = f(p_0(n), n) = 2 + (2n-3)\log\left(\frac{8n-12}{8n-9}\right)$$

and we have $f(p,n) \geq g(n)$ for all $p \in [0,1]$.

Let us prove that g in increasing on $[2, \infty)$.

$$\begin{aligned} \ln(2)\, g'(n) &= 2\ln\left(\frac{8n-12}{8n-9}\right) + 8\frac{2n-3}{8n-12} - 8\frac{2n-3}{8n-9} \\ &= 2\ln(4) + 2 + 2\ln(x) - 8x \\ \text{where} \quad x &= \frac{2n-3}{8n-9}. \end{aligned}$$

Note that $0 \leq x < 1/4$ for all $n \geq 2$. The function $2\ln(x) - 8x$ has derivative $2/x - 8 \geq 0$ when x is is $(0, 1/4]$. It follows that it is an increasing function on

$(0, 1/4]$. It follows that $2\ln(x) - 8x \leq 2\ln(1/4) - 8(1/4) = -2\ln(4) - 2$ for all x in $(0, 1/4]$. Therefore,

$$\ln(2)\, g'(n) \leq 0$$

for all $n \geq 2$. If we show that $g(n)$ has a limit $l \geq 0.91$ when $n \to \infty$, it will follow that $g(n)$ (and therefore $f(p, n)$) is bounded by this limit, which will complete the proof of the proposition. We have

$$\begin{aligned} g(n) &= 2 + \frac{2n-3}{\ln 2} \ln\left(1 - \frac{3}{8n-9}\right) \\ &\to 2 - 6/(8\ln 2) > 0.91 \text{ when } n \to \infty \end{aligned}$$

because

$$\begin{aligned} (2n-3)\ln\left(1 - \frac{3}{8n-9}\right) &\sim (2n-3)\left(-\frac{3}{8n-9}\right) \\ &\sim -2n\,\frac{3}{8n} \to -6/8. \end{aligned}$$

This completes the proof. ■

Proposition 3 *For all $k \geq 1$, $n \geq 2$ we have*

$$\begin{aligned} & 2k + \log(3)L - L\log\left(\frac{L}{(2n-3)k}\right) \\ & - ((2n-3)k - L)\log\left(\frac{(2n-3)k - L}{(2n-3)k}\right) - 2k - \log\left(3^L \binom{(2n-3)k}{L}\right) \\ \geq\ & -\log(2n-3) - \log k - 2 \end{aligned}$$

Proof. First note that, by Stirling's inequality,

$$\begin{aligned} \log\left(3^L \binom{(2n-3)k}{L}\right) &< \frac{\log_3 3^L}{\log_3 2} + ((2n-3)k+1)\log((2n-3)k+1) \\ &\quad - \frac{(2n-3)k}{\ln 2} - \Bigg(L\log L - \frac{L}{\ln 2} \\ &\quad + ((2n-3)k - L)\log((2n-3)k - L) - \frac{(2n-3)k - L}{\ln 2}\Bigg) \\ &= L\log 3 + ((2n-3)k+1)\log((2n-3)k+1) \\ &\quad - L\log L - ((2n-3)k - L)\log((2n-3)k - L). \end{aligned}$$

Then,

$$
\begin{aligned}
& 2k + L\log 3 - L\log\left(\frac{L}{(2n-3)k}\right) - ((2n-3)k-L)\log\left(\frac{(2n-3)k-L}{(2n-3)k}\right) \\
& -2k - \log\left(3^L \binom{(2n-3)k}{L}\right) \\
> \quad & -L\log\left(\frac{L}{(2n-3)k}\right) - ((2n-3)k-L)\log\left(\frac{(2n-3)k-L}{(2n-3)k}\right) \\
& -((2n-3)k+1)\log((2n-3)k+1) + L\log L \\
& +((2n-3)k-L)\log((2n-3)k-L) \\
= \quad & -L\log L + L\log((2n-3)k) - ((2n-3)k-L)\log((2n-3)k-L) \\
& +((2n-3)k-L)\log((2n-3)k) - ((2n-3)k+1)\log((2n-3)k+1) \\
& +L\log L + ((2n-3)k-L)\log((2n-3)k-L) \\
= \quad & L\log((2n-3)k) + ((2n-3)k-L)\log((2n-3)k) \\
& -((2n-3)k+1)\log((2n-3)k+1)
\end{aligned}
$$

$$
\begin{aligned}
= \quad & ((2n-3)k)\log((2n-3)k) - ((2n-3)k+1)\log((2n-3)k+1) \\
= \quad & ((2n-3)k)\log((2n-3)k) \\
& -((2n-3)k+1)\left(\log\left(1+\frac{1}{(2n-3)k}\right) + \log((2n-3)k)\right) \\
\geq \quad & ((2n-3)k)\log((2n-3)k) \\
& -((2n-3)k+1)\left(\log((2n-3)k) + \frac{1}{(2n-3)k}\right) \\
= \quad & ((2n-3)k)\log((2n-3)k) \\
& -((2n-3)k+1)\log((2n-3)k) - \frac{(2n-3)k+1}{(2n-3)k} \\
= \quad & -\log((2n-3)k) - 1 - \frac{1}{(2n-3)k} \\
\geq \quad & -\log(2n-3) - \log k - 2
\end{aligned}
$$

This completes the proof. ■

Proposition 4 *For all $k \geq 7$, $n \geq 2$ and $L \leq (2n-3)k$ we have*

$$
\begin{aligned}
& 4k + (2 + \lceil\log(2n-3)\rceil)L + \log(2n-3) \\
& -2k - \left(\lceil\log 3^L\rceil + \left\lceil\log\binom{(2n-3)k}{L}\right\rceil\right) \quad \geq \quad 0
\end{aligned}
$$

Proof. First note that

$$\begin{aligned} & 4k + (2 + \lceil \log(2n-3) \rceil)L + \log(2n-3) \\ & -2k - \left(\lceil \log 3^L \rceil + \left\lceil \log \binom{(2n-3)k}{L} \right\rceil \right) \\ > \; & 4k + (2 + \lceil \log(2n-3) \rceil)L + \log(2n-3) \\ & -2k - \log 3^L - \log \binom{(2n-3)k}{L} - 2 \\ = \; & 4k + (2 + \lceil \log(2n-3) \rceil)L + \log(2n-3) \\ & -2k - \log \left(3^L \binom{(2n-3)k}{L} \right) - 2. \end{aligned}$$

If we add and subtract the term

$$2k + \log(3)L - L \log \left(\frac{L}{(2n-3)k} \right) - ((2n-3)k - L) \log \left(\frac{(2n-3)k - L}{(2n-3)k} \right),$$

then by Propositions 2 and 3 we have that

$$\begin{aligned} & 4k + (2 + \log(2n-3))L + \log(2n-3) - 2k - \log \left(3^L \binom{(2n-3)k}{L} \right) - 2 \\ \geq \; & 0.91k - \log(2n-3) - \log k - 2 + \log(2n-3) - 2 \\ = \; & 0.91k - \log k - 4 \\ > \; & -0.44 \end{aligned}$$

when $k \geq 7$. Therefore,

$$\begin{aligned} & 4k + (2 + \lceil \log(2n-3) \rceil)L + \log(2n-3) \\ & -2k - \left(\lceil \log 3^L \rceil + \left\lceil \log \binom{(2n-3)k}{L} \right\rceil \right) \quad > \quad -0.44 \end{aligned}$$

but since the left hand side is always integer, we then have

$$\begin{aligned} & 4k + (2 + \lceil \log(2n-3) \rceil)L + \log(2n-3) \\ & -2k - \left(\lceil \log 3^L \rceil + \left\lceil \log \binom{(2n-3)k}{L} \right\rceil \right) \quad \geq \quad 0, \end{aligned}$$

as required. ■